Complex instruction set computing architecture for performing accurate quantum Z rotations with less magic

Andrew J. Landahl[1, 2, 3, *] and Chris Cesare[2, 3, †]

[1] *Advanced Device Technologies, Sandia National Laboratories, Albuquerque, NM, 87185, USA*
[2] *Center for Quantum Information and Control, University of New Mexico, Albuquerque, NM, 87131, USA*
[3] *Department of Physics and Astronomy, University of New Mexico, Albuquerque, NM, 87131, USA*

We present quantum protocols for executing arbitrarily accurate $\pi/2^k$ rotations of a qubit about its Z axis. Reduced instruction set computing (RISC) architectures typically restrict the instruction set to stabilizer operations and a single non-stabilizer operation, such as preparation of a "magic" state from which $T = Z(\pi/4)$ gates can be teleported. Although the overhead required to distill high-fidelity copies of this magic state is high, the subsequent quantum compiling overhead to realize Z rotations in a RISC architecture can be much greater. We develop a complex instruction set computing (CISC) architecture whose instruction set includes stabilizer operations and preparation of magic states from which $Z(\pi/2^k)$ gates can be teleported, for $2 \leq k \leq k_{\max}$. This results in a substantial overall reduction in the number of gates required to achieve a desired gate accuracy for Z rotations. The key to our construction is a family of shortened quantum Reed-Muller codes of length $2^{k+2} - 1$, whose magic-state distillation threshold shrinks with k but is greater than 0.85% for $k \leq 6$.

PACS numbers: 03.67.Lx

I. INTRODUCTION

One of the biggest challenges in quantum information science is that quantum information is incredibly fragile. Even with great experimental care, decoherence can quickly corrupt key features such as superposition and entanglement. To circumvent the ravages of decoherence, one can consider alternative models of quantum computation, such as adiabatic quantum computation [1–3], which may offer direct physical immunity to certain classes of noise [4–14]. Another approach is to encode quantum information redundantly in an error-correcting code and process it fault-tolerantly to suppress the catastrophic propagation of errors [15, 16]. Somewhat miraculously, this latter approach works, and works arbitrarily well, when quantum computations are expressed as quantum circuits in which each elementary operation has a failure probability below a value known as the *accuracy threshold* [17–23]. Estimates for the accuracy threshold vary, and depend in part on the specifics of the fault-tolerant quantum computing protocol used. One of the more favorable estimates is $\approx 1\%$ for a protocol based on Kitaev's surface codes [24–27]. An outstanding grand challenge in quantum information science is finding a way to marry fault-tolerance methods with intrinsically robust computational models to achieve fault tolerance with more achievable resource requirements [28–31].

One of the factors driving up the resource requirements in fault-tolerant quantum computing is the need to restrict the set of elementary operations in the "primitive" or "physical" instruction set to be finite. This is necessary because these instructions are presumed to be implementable only up to some maximal accuracy. One of the main jobs of a fault-tolerant quantum computing protocol is to define how one should sequence these primitive instructions together to synthesize arbitrarily accurate versions of each element of a universal "encoded" or "logical" instruction set, even when the primitive instructions themselves are faulty. Then, using these logical instructions, one can realize any quantum algorithm arbitrarily reliably, even in the face of decoherence and other sources of noise.

In a typical fault-tolerant quantum computing protocol, some logical instructions are "easy" to synthesize in that their error is solely a function of the errors in the primitive instructions from which they are composed. The accuracy of these logical instructions can be improved arbitrarily well by using arbitrarily good quantum codes. More quantitatively, the number of gates and qubits required to achieve approximation error ϵ for the "easy" instructions scales as $\mathcal{O}(\log^\alpha(1/\epsilon))$, where α depends on the protocol, predominantly on the quantum code and classical decoding algorithm it uses. Standard techniques for realizing such gates include transversal action [22, 23] and code deformation [25, 26]. 2D topological codes using most-likely-error decoding can achieve $\alpha = 3$ [25, 26]; Pippenger has conjectured that it should be possible to lower α all the way to 1 [32].

Most protocols also have a set of logical instructions that are "hard" to synthesize, requiring additional methods and resources. The Eastin-Knill theorem, for example, guarantees that no protocol can realize a universal logical instruction set by transversal action alone [33]. A typical approach to synthesizing these hard logical instructions is to use the "magic state" approach, in which the "hard" instructions are state preparations that are distilled to high fidelity using the "easy" operations

*alandahl@sandia.gov
†ccesare@unm.edu

[34]. The number of ideal gates and qubits required to achieve approximation error ϵ in this approach scales as $\mathcal{O}(\log^{\beta}(1/\epsilon))$, where β depends on the magic-state distillation protocol. When the the resource costs for the "easy" gates are also considered, the combined overhead scales as $\mathcal{O}(\log^{\alpha+\beta}(1/\epsilon))$. In the original Bravyi-Kitaev 15-to-1 distillation protocol [34], $\beta = \log_3 15 \approx 2.47$. More recent constructions by Bravyi and Haah [35] and by Jones [36] achieve $\beta = \log_2 3 \approx 1.58$. Bravyi and Haah conjecture that it should be possible to lower β all the way to 1 [35].

As an aside, it is worth mentioning that fault-tolerant quantum computing protocols based on some quantum codes have no "hard" logical instructions at all. For example, the 3D (and higher-dimensional) topological color codes have this feature [37, 38]. They cleverly circumvent the Eastin-Knill theorem by making (non-transversal!) quantum error correction be the process by which magic-states are prepared. A challenge to using these codes in practice is that implementing them without relying on long-distance quantum communication requires 3D spatial geometry, but many quantum technologies are naturally restricted to 1D or 2D. Even more challenging is that the only explicit 3D color code of which we are aware is the 15-qubit shortened quantum Reed-Muller code [37].

Because of the additional overhead incurred in synthesizing "hard" logical instructions, research to date has focused on what one might term *reduced instruction set computing*, or RISC architectures in which only a single "hard" logical instruction is added to an otherwise "easy" logical instruction set. However, in the big picture, the logical instructions are intended to be used to execute quantum algorithms, and constraining oneself to a RISC architecture is not necessarily the wisest choice for reducing the overall qubit and gate overhead. In order to compile the logical instructions into a sequence that approximates a quantum computation with error at most ϵ, one must use $\mathcal{O}(\log^{\gamma}(1/\epsilon))$ gates, where γ depends on the quantum compiling algorithm used. The overall cost of fault-tolerantly implementing a quantum computation is then $\mathcal{O}(\log^{\alpha+\beta+\gamma}(1/\epsilon))$. By increasing the size of the instruction set so that one has a *complex instruction set computing*, or CISC architecture, one can optimize both β and γ together rather than separately. When quantum compiling is optimized independently, γ can be no lower than 1 [39], a value recently achieved by an explicit Diophantine-equation-based algorithm by Kliuchnikov *et al.* [40]. For comparison's sake, the more well-studied Dawson-Nielsen variant of the Solovay-Kitaev algorithm achieves $\gamma = \log 5/\log(3/2) \approx 3.97$ [41].

To compare and contrast the RISC and CISC approaches more concretely without being encumbered by details of quantum error correcting codes and fault tolerance (which only contribute to α and a delineation of which logical instructions are "easy" or "hard"—properties shared by both approaches), we abstract these details away and simply consider the straightforward problem of how to approximate $\pi/2^k$ rotations of a qubit about its Z axis with a desired error at most ϵ' when we are given the ability to perform a proscribed set of "easy" instructions that are error-free and a proscribed set of "hard" instructions that have error at most $\epsilon > \epsilon'$. In this setting, it is clear that some kind of distillation of the hard instructions will be necessary to synthesize the Z rotations with lower error. $Z(\pi/2^k)$ rotations are a natural candidate transformation to use to compare RISC and CISC approaches, because they arise in many quantum algorithms, for example those that make use of the quantum Fourier transform [42].

In Sec. II, we formulate the statement of the problem we are considering more precisely. In Sec. III, we review the standard RISC solution to this problem. In Sec. IV, we describe our CISC solution, and compare it to the RISC solution, demonstrating that our solution offers a substantial reduction in the number of gates used to achieve this task. Sec. V concludes. Appendix A elaborates the shortened quantum Reed-Muller codes we use to effect our protocol, and Appendix B formulates a testable set of criteria one can use to check if a code admits $Z(\pi/2^k)$ transversally.

II. PROBLEM STATEMENT

Consider quantum Z rotations of the form

$$Z_k := \begin{pmatrix} 1 & 0 \\ 0 & e^{i\pi/2^k} \end{pmatrix} = e^{i\pi/2^{k+1}} R_z\left(\frac{\pi}{2^k}\right), \tag{1}$$

for integers $k \geq 0$. As a shorthand, we use Z to denote the Pauli operator Z_0 and S and T to denote the rotations Z_1 and Z_2 respectively. We are interested in the scenario in which the Z_k gates are not available directly, but rather their action on $|+\rangle$ states is, where $|+\rangle := H|0\rangle = (|0\rangle + |1\rangle)/\sqrt{2}$ and $H := (X+Z)/\sqrt{2}$. For concreteness, let $\mathcal{Z}_{k_{\max}}$ denote the set of states of the form

$$Z_k|+\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle + e^{i\pi/2^k}|1\rangle\right) \tag{2}$$

for $2 \leq k \leq k_{\max}$.

In conjunction with the set $\mathcal{S}$ of *stabilizer operations* [43], the set $\mathcal{Z}_{k_{\max}}$ can effect universal quantum computation, even when restricted to $k_{\max} \leq 2$ [42]. We are interested in the scenario in which a certain (overcomplete) generating set for $\mathcal{S}$ is available, namely the set consisting of the operations

$$\{I,\ X,\ Y,\ Z,\ S,\ S^\dagger,\ H\} \cup \{|0\rangle,\ |+\rangle,\ M_Z,\ M_X\} \tag{3}$$

and

$$\{\Lambda(X^{q_1} \otimes \cdots \otimes X^{q_m}) \mid q_i \in \{0,1\}\}, \tag{4}$$

where I, X, Y, and Z denote the Pauli operators, M_X and M_Z denote projective measurements in the X and Z bases (but which may be "destructive" in that they do

not necessarily prepare X or Z eigenstates after the measurement), and $\Lambda(X^q)$ denotes the one-control, many-target controlled-NOT gate, where the number of targets m is some efficiently computable number. The unitary gates in this generating set generate a subgroup of the stabilizer operations known as the *Clifford* operations [43], which are the set of operations that conjugate (tensor products of) Pauli operators to (tensor products of) Pauli operators.

These generators of $\mathcal{S}$ are "easy" to perform at the logical level for the 4.8.8 2D color codes, motivating our choice [38]. The set is also almost "easy" for Kitaev's 2D surface codes [24], except generating S and $S^\dagger$ requires some constant startup costs that can be amortized [44]. Amazingly, as alluded to in the introduction, all elements from the set $\mathcal{S} \cup \mathcal{Z}_2$—a universal set—are "easy" to perform at the logical level for 3D color codes, but 3D geometries are required to realize error correction with these codes in a spatially local manner [38].

While errors in the "easy" operations can be suppressed arbitrarily low using 2D topological codes, errors in the operations in $\mathcal{Z}_{k_{\max}}$ cannot, making these operations "hard" for these codes. The states in $\mathcal{Z}_{k_{\max}}$ can be "injected" into such codes at the logical level [26], but doing so also injects the errors in the state. In other words, if the states in $\mathcal{Z}_{k_{\max}}$ have errors that are at most ϵ (as measured by the trace distance [42]) as primitive instructions, then the injected states will have errors that are essentially the same when they become logical instructions, assuming the injection process itself adds errors at a low enough probability [84].

Motivated by these properties of 2D topological codes, we will fix the control model for our study to be the aforementioned generators of $\mathcal{S}$ and $\mathcal{Z}_{k_{\max}}$, and the error model to be one in which the operations in $\mathcal{S}$ are error-free but in which the $Z_k|+\rangle$ states in $\mathcal{Z}_{k_{\max}}$ each err by at most ϵ, as measured by the trace distance. Notice that this control model makes no reference to codes or fault-tolerant quantum computing protocols. We have abstracted these away to focus on how to combine elementary operations in $\mathcal{S}$ and $\mathcal{Z}_{k_{\max}}$ to achieve high-fidelity Z rotations.

The question we address here is,

> *How many elementary quantum operations from $\mathcal{S}$ and $\mathcal{Z}_{k_{max}}$ does it take to approximate Z_k with error at most $\epsilon' < \epsilon$ as a function of k_{max}, k, ϵ, and ϵ'?*

The values of k we are interested in could be smaller than, equal to, or larger than $k_{\max}$. However, since Z_0 and Z_1 are both in the error-free set $\mathcal{S}$, we are only interested in $k \geq 2$.

III. TRADITIONAL QUANTUM RISC ARCHITECTURE SOLUTION

The standard method for refining the accuracy of a Z_k rotation in this model is to synthesize it with what one might term a quantum *reduced instruction set computing*, or quantum RISC, architecture. The main idea is to only synthesize $T := Z_2$ gates to high accuracy and then rely on a quantum compiling algorithm to approximate Z_k arbitrarily well with a quantum circuit over T gates and adaptive stabilizer operations. The overall process can be broken into the three steps of *quantum compiling*, *quantum gate teleportation*, and *magic-state distillation*.

A. Protocol

1. Quantum compiling

The first step, *quantum compiling*, generates a classical description of an ideal quantum circuit that approximates Z_k to accuracy $\epsilon_{\rm qc}$ using $\mathcal{O}(\log^\gamma(1/\epsilon_{\rm qc}))$ quantum operations drawn from some instruction set, for some small constant γ. While the error $\epsilon_{\rm qc}$ can be measured in multiple ways, a wise choice is to measure $\epsilon_{\rm qc}$ using the completely-bounded ("diamond") trace distance [19, 45, 46] for reasons that we will explain later. Examples of quantum compiling algorithms include the Solovay-Kitaev algorithm [19, 39, 41, 42, 47–49], the Kitaev phase kickback algorithm [50–52], programmed ancilla algorithms [36, 53, 54], genetic algorithms [55], and even Diophantine-equation algorithms [40]. When the accuracy demand is not great, it is sometimes even plausible to use algorithms which take exponential time to find very short approximation sequences [56–59]. As noted in the introduction, values for γ range from 3.97 to 1.

Quantum compiling algorithms typically assume that the elements of the instruction set are error-free. If one implements the compiled circuit $Z_k^{(\rm qc)}$ for Z_k with operations that may be in error, the resulting approximation error will increase. To calculate the total error ϵ_k in this flawed circuit $\tilde{Z}_k^{(\rm qc)}$, we use the fact that the diamond norm has many useful properties, including obeying the triangle inequality, the chaining inequality, and unitary invariance [60]. Using these, we can bound ϵ_k as

$$\begin{aligned} \epsilon_k &= d_\diamond\Big(Z_k, \tilde{Z}_k^{(\rm qc)}\Big) && (5)\\ &\leq\ d_\diamond\Big(Z_k, Z_k^{(\rm qc)}\Big) + d_\diamond\Big(Z_k^{(\rm qc)}, \tilde{Z}_k^{(\rm qc)}\Big) && (6)\\ &\leq \epsilon_{\rm qc} + n_T\epsilon_T, && (7)\end{aligned}$$

where the compiled circuit uses n_T T gates, each with error at most ϵ_T. To achieve the desired approximation error of ϵ', it follows that sufficient conditions are

$$\begin{aligned} \epsilon_{\rm qc} &\leq C_{\rm qc}\epsilon' && (8)\\ \epsilon_T &\leq C_T\epsilon'/n_T, && (9)\end{aligned}$$

for positive constants constrained to obey

$$C_T + C_{\rm qc} \leq 1. \tag{10}$$

2. Quantum gate teleportation

The second step, *quantum gate teleportation*, replaces each T gate in the quantum-compiled circuit by an adaptive stabilizer circuit that teleports the T gate from the state $T|+\rangle$ or $T^\dagger|+\rangle$ to the desired qubit. An example of a teleportation circuit using $T|+\rangle$ is depicted in Fig. 1. The circuit is also correct if both T operators are changed to $T^\dagger$; it is even correct if only one of the T operators is changed to a $T^\dagger$ if the classical control is also changed to act on a 0 instead of a 1.

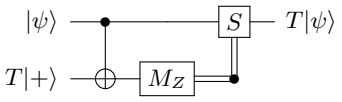

FIG. 1: Circuit for teleporting the T gate from the $T|+\rangle$ magic state.

It is worth remarking that the classically controlled S gate in this circuit is not a Clifford gate because of its adaptive nature; it is a common misconception that the adaptive S gate is a Clifford gate. While the fact that an adaptive S gate is non-Clifford follows mathematically from the fact that a bit flip on the classical control propagates to a non-Pauli operator, it is perhaps more amusing and convincing to consider the following consequence if the adaptive S gate had been a Clifford gate. Suppose our original circuit was a "programmable" circuit in which m of the qubits specified the quantum circuit to be implemented and n of the qubits described the initial state of the qubits it was supposed to act on. Without loss of generality, the state of these $m+n$ qubits are describable by a classical $(m+n)$-bit string. Suppose further that we replaced each T gate in the circuit with the circuit of Fig. 1 and that it had been a Clifford circuit. Finally, suppose we executed this circuit using some quantum device and read out the answer, after which the quantum device that executed the circuit was irrevocably destroyed. No problem! To run the circuit on any other $(m+n)$-bit input string, we can just propagate the Pauli X-flips on the input through to a set of Pauli X flips on the answer we read out, obtaining the output on the new input. The propagation is guaranteed to be efficiently implementable by any of a variety of explicit algorithms derived from the Gottesman-Knill theorem [43, 61, 62]. By this logic, there would never be a need to run a quantum algorithm more than once on a given instance size, since the output of the algorithm on all other instances of the same size would be efficiently computable from it with classical post-processing. Even more amazingly, running the quantum algorithm once is sufficient to determine the output of any other quantum algorithm requiring the same number of bits for specification! Quantum computers are great, but not that great—the adaptive S gate is not a Clifford operation.

Each teleportation circuit, not counting preparation of its inputs, adds one $\Lambda(X)$ gate, one I operation, one M_Z measurement, and half of the time, one S operation, for a grand total of 3.5 operations on average and 4 operations in the worst case. Since teleportation is only performed once per T gate, we use the average value of 3.5.

3. Magic-state distillation

The third step, *magic-state distillation*, generates $T|+\rangle$ or $T^\dagger|+\rangle$ states with accuracy ϵ_T from a much larger collection of states whose accuracy is only ϵ. Reichardt showed that this is possible using an *ideal* (error-free) stabilizer circuit if and only if ϵ is less than the distillation threshold $(2-\sqrt{2})/4 \approx 0.146$ [63]. When operations in the stabilizer circuit can err, the evaluation of the threshold is more complex, as studied by Jochym-O'Connor *et al.* [64].

There are multiple variations on how to implement magic-state distillation discussed in the literature [34, 35, 65–68]; a popular one is the 15-to-1 Bravyi-Kitaev protocol [34] based on the 15-qubit shortened quantum Reed-Muller code $\overline{QRM}(1,4)$. (See Appendix A for an explanation of this notation.)

In their original paper, Bravyi and Kitaev proposed the following distillation protocol:

1. Prepare the state $(T|+\rangle)^{\otimes 15}$.
2. Apply $A := TXT^\dagger \sim SX$ with probability 1/2 on each qubit.
3. Measure the Z checks for $\overline{QRM}(1,4)$.
4. Identify qubits to flip to reset the Z-check syndrome to 0.
5. Apply A to the identified qubits.
6. Measure the X checks for $\overline{QRM}(1,4)$.
7. Declare failure if the X-check syndrome is not 0. Otherwise, proceed to the next step.
8. Apply the coherent decoding circuit for $\overline{QRM}(1,4)$. The output is $T^\dagger|+\rangle$ with a higher fidelity.

Because this algorithm uses many gates and qubits for syndrome measurement and correction, attempts have been made to simplify the circuit. For example, the X-check measurements can be pushed through the decoding circuit to become simpler individual M_X measurements. Some have attempted to push the Z-check measurements and A corrections through the circuit as well. Because the A operators are not Pauli operators, their propagation through the Clifford decoding circuit is complicated.

The equivalent post-decoding-circuit Z-check measurements become elaborate entangled measurements that cannot be reconstructed by simply measuring each qubit individually, despite suggestive circuits drawn in the literature [27, 69].

A clever alternative by Raussendorf *et al.* obviates the need for measuring complicated operators by encoding half of a Bell state and processing it to distill the output on the other half [26]. Their protocol is as follows:

1. Prepare the state $|\Phi^+\rangle := (|00\rangle + |11\rangle)/\sqrt{2}$.
2. Send half of $|\Phi^+\rangle$ through the coherent encoding circuit for $\overline{QRM}(1,4)$.
3. Prepare the state $(T|+\rangle)^{\otimes 15}$.
4. Apply A with probability $1/2$ on each of these 15 qubits.
5. Teleport the T gate from each of these "twirled" $T|+\rangle$ states to a corresponding qubit on the encoded half of $|\Phi^+\rangle$.
6. Measure M_X on each of the 15 encoded qubits.
7. Infer the X-check values from the appropriate products of these M_X measurements. Declare failure if the X-check syndrome is not 0. Otherwise, proceed to the next step.
8. Infer the logical X value for $\overline{QRM}(1,4)$ by taking the product of all of the M_X values. If it is -1, apply Z to the other half of the original Bell state. The unmeasured half of the original Bell state is $T^\dagger|+\rangle$ with a higher fidelity.

The inclusion of the twirl operation is omitted by Raussendorf *et al.* and by many others who have built upon this protocol [70, 71]. However, twirling is essential in the analysis by Bravyi and Kitaev in deriving Eq. (11) below for the accuracy of the distilled output state:

$$\epsilon_{\text{out}}(\epsilon) = \frac{1-(1-2\epsilon)^7(30\epsilon+(1-2\epsilon)^8)}{2(1+15(1-2\epsilon)^8)} \tag{11}$$

$$\approx 35\epsilon^3. \tag{12}$$

That said, Jochym-O'Connor *et al.* have discovered that magic-state distillation works at least as well, and maybe even better, when twirling is omitted in a distillation protocol based on the five-qubit code [64]. Inspired by this result, we too omit the twirling circuits from the protocol yet still use the Bravyi-Kitaev formula as a sort of loose guideline for how much the fidelity has improved, expecting that the fidelity increase may even be better.

A quantum circuit that implements the Raussendorf *et al.* protocol is depicted in Fig. 2.

The final Z correction from step 8 that may need to be applied is not depicted in Fig. 2 because it can be incorporated into the subsequent teleportation circuit in Fig. 1 instead at a lower gate cost. To do this, we just replace the I gate there with a Z gate, as needed. Alternatively, we could keep the I gate and replace the S gate with $SZ = S^\dagger$. Either way, the number of gates in the teleportation circuit is 3.5 on average and 4 in the worst case.

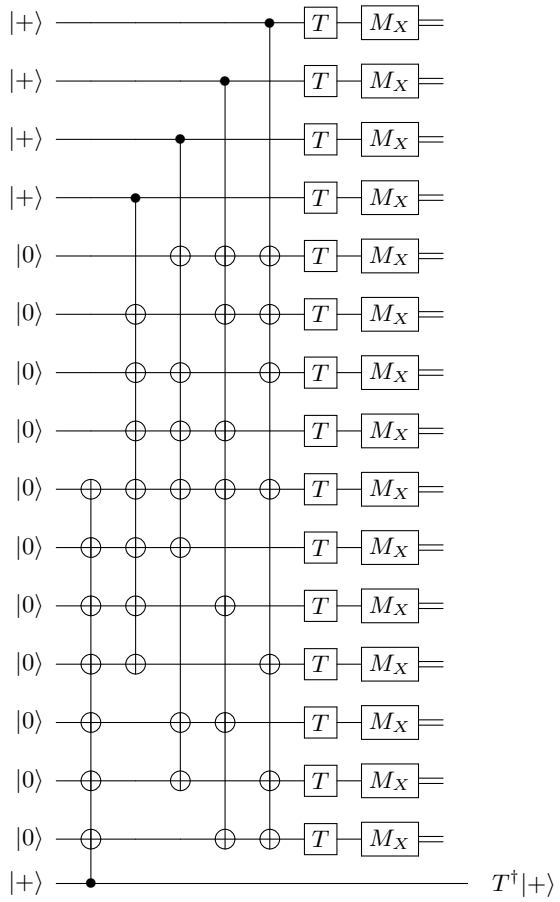

FIG. 2: Distillation circuit for $T^\dagger|+\rangle$ states, adapted from Ref. [26]. After the Bell state preparation, the gates perform the coherent encoding circuit for the 15-qubit shortened quantum Reed-Muller code. (See Appendix A for details.) The T gates are performed by gate teleportation, using the circuit from Fig. 1. This distillation circuit also distills $T|+\rangle$ states on $T^\dagger|+\rangle$ inputs.

Let us now count the number of gates used by the Raussendorf *et al.* protocol.

To achieve $\epsilon_{\text{out}} \leq \epsilon_T$, one iterates this distillation process ℓ times, where

$$\ell(\epsilon_T, \epsilon) = \left\lceil \frac{\log \epsilon_T}{\log \epsilon_{\text{out}}(\epsilon)} \right\rceil. \tag{13}$$

In addition to this, each of these rounds must themselves be repeated t times because the X checks may fail to give a trivial syndrome. Because low-error states decode with higher probability, the expected number of repetitions is small when ϵ is small. Specifically, the ex-

pected number of times each round must be repeated is

$$\mathbb{E}[t(\epsilon)] = \frac{16}{1 + 15(1-2\epsilon)^8}. \tag{14}$$

We can compress identity gates out of each "wire" of the distillation circuit by moving each preparation as far to the right as possible and each T/M_X pair as far to the left as possible. While this by-hand compression of the distillation circuit is not necessarily optimal (a different choice of generators could be used, for example), it reduces the number of stabilizer operations in the circuit of Fig. 2 substantially. The number of gates occurring before the T gates is compressed to 35—16 preparations, 5 $\Lambda(X^q)$ gates, 10 identity gates interior to the $\Lambda(X^q)$ gates, and 4 identity gates on the bottom qubit. Each T/M_X pair requires 5.5 operations on average (a count which includes the preparation of the $T|+\rangle$ state in the teleportation circuit of Fig. 1), adding 82.5 gates on average. The bottom qubit must idle during this, incurring 1 identity gate during the final M_X measurements and 3 identity gates during the T teleportations before that, as it is exceptionally likely (probability $1 - 2^{-15}$) that at least one T teleportation circuit will require a corrective step. This leads to a grand total of approximately 121.5 expected gates on average.

We could reduce the number of qubits used from 31 to 17 by reusing the same qubit for each teleportation circuit. As noted in Ref. [69], we could further reduce the number of qubits by reusing the same qubit for each of the first four control lines, bringing the qubit total down to 14. However, each of these modifications introduces additional qubit idling in the form of extra identity gates. Since we are interested in reducing the number of gates, we do not pursue these optimizations.

B. Resource analysis

As mentioned in the introduction, asymptotically the total number of operations required to approximate a Z_k gate with error ϵ' is $\mathcal{O}(\log^{\alpha+\beta+\gamma}(1/\epsilon'))$, where the exponents describe various overheads of the steps involved: fault-tolerant stabilizer operations (α), magic-state distillation (β), and quantum compiling (γ). While a good starting point, asymptotic analysis like this fails to convey the great number of elementary operations needed to implement Z_k gates, as it sweeps the (large!) constants under the rug. The explicit expression for the expected number of gates used by the RISC approach to approximate Z_k to error ϵ' using $T|+\rangle$ states whose error is ϵ is

$$\begin{aligned} n_{\text{gates}}^{\text{RISC}}(Z_k, \epsilon', \epsilon) \leq\ & n_S^{(\text{qc})}(Z_k, C_{\text{qc}}\epsilon') + n_T^{(\text{qc})}(Z_k, C_{\text{qc}}\epsilon') \\ & \times n_T^{\text{RISC}}\left(C_T\epsilon'/n_T^{(\text{qc})}(Z_k, C_{\text{qc}}\epsilon'), \epsilon\right), \end{aligned} \tag{15}$$

$$n_T^{\text{RISC}}(\tilde{\epsilon}, \epsilon) = (121.5)\mathbb{E}[t(\epsilon)]\ell(\tilde{\epsilon}, \epsilon) + 3.75, \tag{16}$$

where $n_S^{(\text{qc})}(Z_k, \tilde{\epsilon})$ and $n_T^{(\text{qc})}(Z_k, \tilde{\epsilon})$ denote the number of stabilizer operations and T gates found by some quantum compiling algorithm that approximates Z_k to within $\tilde{\epsilon}$.

To better appreciate the compiling resources needed, we consider the case when $C_{\text{qc}} = C_T = 1/2$, which balances the quality demands of quantum compiling and magic-state distillation. Let us also give the T gate a generous error rate of $\epsilon = 10^{-4}$, which is well below the estimated threshold of $\approx 1\%$ for fault-tolerant quantum computation with surface codes [26, 27]. To generate values for $n_T^{(\text{qc})}$ and $n_S^{(\text{qc})}$, we appeal to the results by Kliuchnikov *et al.* [58], who use a streamlined version of the Dawson-Nielsen compiling algorithm iterated n_{DN} times to achieve a given quantum compiling error. They exhaustively tabulated gate counts on their website http://qcirc.iqc.uwaterloo.ca/ for k up to 29. The number of operations $n_{\text{gates}}^{\text{RISC}}$ required to synthesize Z_k with these parameters to various approximation levels and for small values of k are listed in Table I.

IV. QUANTUM CISC ARCHITECTURE SOLUTION

After witnessing just how many gates are required to implement Z_k rotations using a quantum RISC architecture, it's natural to ask if extending the instruction set to a quantum *complex instruction set computing* architecture, or quantum CISC architecture, could improve matters. The point is that in any given quantum algorithm instance, one isn't interested in applying *arbitrary* gates but rather a specific set of gates, say Z_k gates up to some maximum value of k in a quantum Fourier transform. Because of this, it may make more sense to just include those gates in the instruction set to begin with rather than compiling them from a more limited instruction set. Even if it is only feasible to include gates up to some value of $Z_{k_{\max}}$, it is reasonable to expect that the quantum compiling task that remains is much easier than if one had restricted oneself to a smaller instruction set.

A. Protocol

Here we consider a programmed-ancilla CISC architecture, in which we pre-compile $Z_k|+\rangle$ states offline that can be used later to teleport the gate Z_k on demand via the circuit in Fig. 3. While the teleportation may require a Z_{k-1} gate for correction, iterating this process recursively is a negative binomial process that converges exponentially quickly—the expected number of Z rotations for any k is two: Z_k on $|+\rangle$ and Z_{k-1} after the measurement. Because of this, to achieve error at most ϵ' on the teleported Z_k gate, the $Z_k|+\rangle$ state and the Z_{k-1} gate need to be performed with errors at most $C_1\epsilon'$ and $C_2\epsilon'$ respectively, where $C_1 + C_2 \leq 1$.

k	n_{DN}	ϵ'	$n_{\mathrm{gates}}^{\mathrm{RISC}}$	$n_{\mathrm{gates}}^{\mathrm{CISC}}$
3	0	3×10^{-3}	3 551	5
	1	9×10^{-5}	16 735	377
	2	1×10^{-6}	84 886	377
	3	6×10^{-10}	815 617	687
	4	7×10^{-15}	3 554 634	748
	5	7×10^{-22}	27 443 509	1 120
	6	9×10^{-33}	170 821 010	1 491
	7	1×10^{-50}	1 244 288 695	2 173
4	0	8×10^{-4}	3 045	6
	1	3×10^{-5}	15 712	901
	2	9×10^{-7}	70 434	901
	3	2×10^{-10}	745 089	1 795
	4	5×10^{-15}	3 490 498	1 795
	5	2×10^{-22}	26 861 004	2 690
	6	2×10^{-33}	167 324 474	4 294
	7	2×10^{-50}	1 235 755 305	6 053
5	0	2×10^{-3}	2 787	6
	1	2×10^{-5}	17 234	2 018
	2	4×10^{-7}	71 453	2 018
	3	2×10^{-10}	766 508	4 030
	4	6×10^{-15}	3 519 361	5 595
	5	3×10^{-22}	26 241 651	7 607
	6	7×10^{-34}	164 504 724	9 974
	7	4×10^{-51}	1 239 288 461	15 548
6	0	2×10^{-3}	3 550	6
	1	2×10^{-5}	17 233	4 701
	2	7×10^{-7}	80 318	8 390
	3	8×10^{-10}	746 097	9 396
	4	3×10^{-14}	3 408 145	13 867
	5	5×10^{-22}	25 181 767	18 561
	6	4×10^{-33}	171 199 899	27 122
	7	5×10^{-51}	1 250 275 506	40 974

TABLE I: Expected number of gates required to approximate Z_k to precision ϵ' derived from the results of Kliuchnikov *et al.* [58], which uses the Dawson-Nielsen RISC architecture with Bravyi-Kitaev 15-to-1 magic-state distillation [34], and the corresponding number of gates required by our CISC architecture. For each number of iterations of the Dawson-Nielsen algorithm n_{DN}, the table lists the smallest output error achievable ϵ', rounded up to its most significant figure, assuming that quantum compiling and magic-state distillation contribute equally to the total error and that the bare error rate of Z_k operations is $\epsilon = 10^{-4}$.

Our CISC approach is distinguished from previous programmed-ancilla approaches [36, 53, 54] in that we distill ancilla $Z_k|+\rangle$ states directly as instructions unto themselves. This is a "top-down" approach in which some of the time auxiliary $Z_{k-1}|+\rangle$ states are needed, and even less of the time $Z_{k-2}|+\rangle$ states are needed, and so on, until we get to the point that very rarely do we need

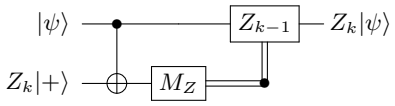

FIG. 3: Magic-state circuit for teleporting the Z_k gate.

$T|+\rangle$ states. The previous approaches are "bottom-up" in that they always compile from $T|+\rangle$ states upwards until the Z_k gate is performed; some of these schemes (notably the recent one by Duclos-Cianci and Svore [54]) reduce resources by including intermediate targets, but ultimately they all start from $T|+\rangle$ preparations at the lowest level. By starting from the top, we avoid the need to probe all the way to the bottom most of the time. As we will see, this results in significant savings in the number of operations needed to synthesize Z_k gates.

The key to our construction is a family of shortened quantum Reed-Muller codes that are defined in Appendix A. The property of these codes that we harness here is that the $\overline{QRM}(1, k+2)$ codes admit the logical Z_k gate *transversally*, namely by applying $Z_k^\dagger$ to each qubit independently. We know this because these codes satisfy the conditions we derived in Appendix B. Because of this transversality property, we can use the $\overline{QRM}(1, k+2)$ code to distill $Z_k^\dagger|+\rangle$ states using a circuit that is essentially the same as the one used in the RISC architecture for distilling T gates. Specifically, if we replace the encoding circuit for $\overline{QRM}(1, 4)$ with the encoding circuit for $\overline{QRM}(1, k+2)$ and replace each T with Z_k in the distillation circuit depicted in Fig. 2, the circuit becomes a distillation circuit for $Z_k^\dagger|+\rangle$ states. As an example, we depict the distillation circuit for $Z_3^\dagger$ in Fig. 4; we derived the encoding circuit for $\overline{QRM}(1, 5)$ in the figure using the methods outlined in Refs. [42, 72]. We defer a proof of why these codes have the transversality property to Appendix B and instead focus on how the protocol works here. We will note here, though, that our proof generalizes the "tri-orthogonality" condition that Bravyi and Haah used to establish the transversality of T gates for their codes to a lemma in coding theory proved by Ward that we call *Ward's Divisibility Test* [73, 74].

Using the $\overline{QRM}(1, k+2)$ code to distill Z_k gates yields the following distillation polynomial, which generalizes Eq. (11):

$$\epsilon_{\mathrm{out}}(\epsilon) = \frac{1 - (1-2\epsilon)^{2^{k+1}-1}\left[2\epsilon(2^{k+2}-1) + (1-2\epsilon)^{2^{k+1}}\right]}{2\left[1 + (2^{k+2}-1)(1-2\epsilon)^{2^{k+1}}\right]} \tag{17}$$

$$\approx \left(1 - 3 \cdot 2^{k+1} + 2^{2k+3}\right)\left(\epsilon^3/3 + \epsilon^4 + \mathcal{O}(\epsilon^5)\right). \tag{18}$$

Approximate values for the distillation threshold for various values of k are listed in Table II; these are the same threshold values one obtains if one used the code to distill Z_{k+1}, but with only a linear, not cubic, distillation polynomial by generalizing the method of Reichardt [63].

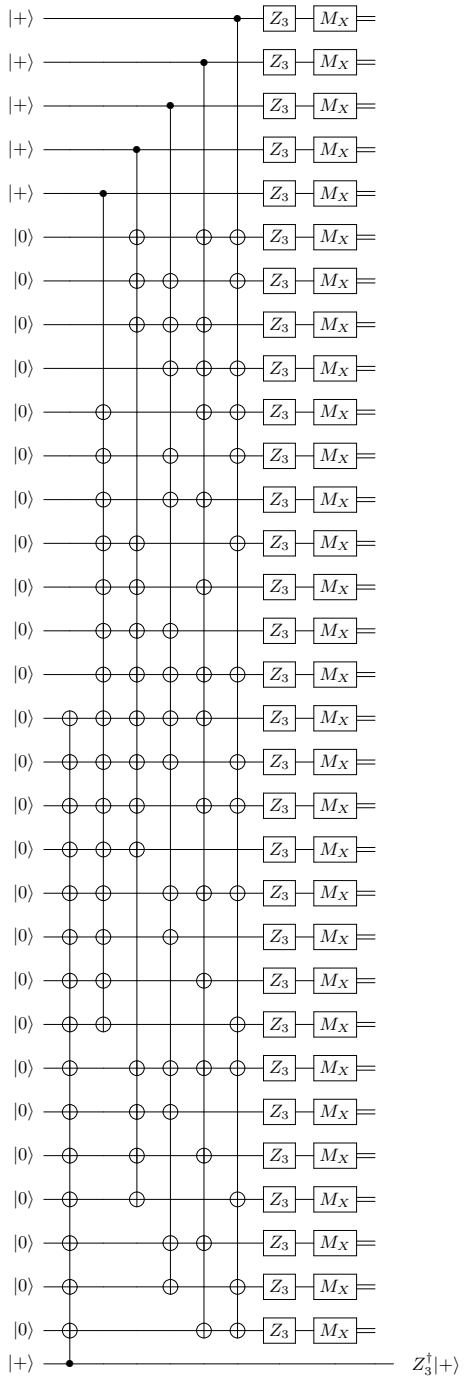

FIG. 4: Distillation circuit for $Z_3^\dagger|+\rangle = \sqrt{T}^\dagger|+\rangle$ states; it is the 31-qubit shortened quantum Reed-Muller code's encoding circuit applied to half of a Bell state followed by the logical Z_3 gate and M_X measurement of the qubits on this encoded half. The Z_3 gates are performed using the teleportation circuit depicted in Fig. 3. This circuit also distills $Z_3|+\rangle$ states on $Z_3^\dagger|+\rangle$ inputs.

Although the distillation threshold drops as k increases, it is still larger than or comparable to the threshold of $\approx 1\%$ for fault-tolerant quantum computation with surface codes [25–27] for values of k less than or equal to 6, where it takes the value $\epsilon_6^{\text{th}} \approx 0.85\%$. This then sets a reasonable upper limit on the size of the complex instruction set one should consider for performing Z_k gates in this way; going further would place greater fidelity demands on the elementary operations than fault-tolerance does.

k	$\epsilon_{\text{out}}/\epsilon^3$	ϵ_k^{th}
2	35	14.15%
3	155	6.94%
4	651	3.44%
5	2 667	1.71%
6	10 795	0.85%
7	43 435	0.43%
8	174 251	0.21%
9	698 027	0.11%
10	2 794 155	0.05%

TABLE II: Distillation polynomials (to most significant order) and distillation thresholds for distilling $|Z_k^\dagger\rangle$ states.

To achieve $\epsilon_{\text{out}} \leq \epsilon'$, one must iterate the distillation circuit

$$\ell(\epsilon', \epsilon) = \left\lceil \frac{\log \epsilon'}{\log \epsilon_{\text{out}}(\epsilon)} \right\rceil \tag{19}$$

times. The expected number of repetitions per iteration needed to achieve distillation success, generalizing Eq. (14), is

$$\mathbb{E}[t(\epsilon)] = \frac{2^{k+2}}{1 + (2^{k+2} - 1)(1 - 2\epsilon)^{2^{k+1}}}. \tag{20}$$

Unlike in the RISC protocol, in which the corrective step in the teleportation circuit added no error, in our protocol each teleportation circuit may add error in its adaptive Z_{k-1} gate. While we could implement the Z_{k-1} gate with low error using our protocol recursively, this adds a great many resources for little benefit. Instead, we simply use a "bare" Z_{k-1} gate that relies on a $Z_{k-1}|+\rangle$ state prepared with error ϵ.

B. Resource analysis

Asymptotically, our CISC protocol achieves a value of $\beta = \beta_k := \log_3(2^{k+2} - 1)$ and $\gamma = 0$. The sum $\beta + \gamma$ is less than the sum of the 15-to-1 Bravyi-Kitaev magic-state distillation β and the Dawson-Nielsen compiling γ for $k \leq 9$. However, since the distillation threshold drops below 0.85% after $k = 6$, as argued earlier, it is probably wisest to stop at $k = 6$. Compared to the best values we know for β (≈ 1.58 by Refs. [35, 36]) and γ (1 by Ref. [40]), our CISC protocol would appear to be only superior for $k \leq 2$. However it is important to remember, as mentioned earlier, that arguing about asymptotics in this way can be very misleading as the constants involved

can be huge. For this reason, as before, we shift our attention to a full accounting the expected number of gates used.

The explicit expression for the expected number of gates used by our CISC approach is just the sum of the number of gates used to distill a $Z_k|+\rangle$ state and the number of gates used to perform the final teleportation:

$$n^{\text{CISC}}_{\text{gates}}(Z_k, \epsilon', \epsilon) = n_{\text{distill}}(Z_k, \epsilon', \epsilon) + n_{\text{teleport}}(Z_k, \epsilon', \epsilon). \quad (21)$$

This is simpler than the corresponding RISC formula (15) in that quantum-compiling factors do not appear, but more complex in that the expressions for each of the terms is more involved.

From Fig. 3, the final teleportation circuit, not counting preparation of its inputs, adds one $\Lambda(X)$ gate, one I operation, one M_Z measurement, and half of the time, one Z_{k-1} operation that must be applied with error at most $C_2\epsilon'$. Additionally, a Z operation may need to be applied to the input $Z_k|+\rangle$ state depending on the outcomes of the M_X measurements in the circuit which distilled $Z_k|+\rangle$; the Z correction will be needed half of the time. We can propagate this through the circuit to the final step, where it acts on the top qubit. By replacing the I gate with the Z gate as we did in the case of $k = 2$, the expected number of gates that teleportation adds is three plus the expected number required for the Z_{k-1} correction:

$$n_{\text{teleport}}(Z_k, \epsilon', \epsilon) = 3 + \frac{1}{2} n^{\text{CISC}}_{\text{gates}}(Z_{k-1}, C_2\epsilon', \epsilon). \quad (22)$$

In the worst case, the factor of 1/2 becomes 1, but we only consider the average case here for the very final teleportation as we did in our RISC analysis.

As with the RISC protocol, we compress the number of gates added by the distillation circuit before counting them. Regardless of any special structure to the $\overline{QRM}(1, k+2)$ code, we know that we can remove the identity gates on the first and last controlled-operations as well as the identity gates on the qubits involved in control lines. The total number of gates in the compressed distillation circuit before the Z_k gates is therefore

$$\begin{aligned} n_{\text{pre } Z_k} &= 2^{k+2} && \text{preparations} \\ &+ (k+3) && \Lambda(X^q) \text{ ops} \\ &+ 2^{k+1}(k+1) + 1 && I \text{ during } \Lambda(X^q) \text{ ops} \\ &- (k+1)^2 && I \text{ on } \Lambda(X^q) \text{ controls} \quad (23) \\ &= 2^{k+1}(k+3) - k^2 - k + 3. && (24) \end{aligned}$$

It is possible to compress the distillation circuit further by taking advantage of structure in the $\overline{QRM}(1, k+2)$ codes arising at individual values of k. For example, the approach just described only compresses the distillation circuit for Z_2 down to 37 gates whereas we were able to compress it down to 35 by examining Fig. 2. The code's special structure allows one to move the leftmost gate on qubit 5 one more step to the right and the rightmost gate on qubit 10 one more step to the left. Similarly, one can shave 10 extra gates off of the gate count for the Z_3 circuit, 32 gates off of the Z_4 circuit, and 84 gates off of the Z_5 circuit. We stopped at Z_5, but such by-hand gate reductions could be continued further. Also, as we did for the Z_2 distillation circuit, we passed up compressions in qubit number that increase the number of gates used.

After the Z_k gates occur, the number of gates is just $n_{\text{post } Z_k} = 2^{k+2}$ for the M_X measurements and the concurrent I gate.

The expression for the number of gates during the teleportation steps for Z_k is not the same as that in Eq. (22). Unlike in that setting, we must explicitly count the preparation of the $Z_k|+\rangle$ states. However, we need not account for a Z correction coming from a distillation circuit because the $Z_k|+\rangle$ states are not distilled. The number of gates that a single Z_k gate requires is therefore a simple recurrence relation that accounts for the four gates in Fig. 3 ($Z_k|+\rangle$, $\Lambda(X)$, I, and M_X) and half of the time a corrective Z_{k-1} gate:

$$n^{(1)}_{\text{tele}}(k) = 4 + \frac{1}{2} n^{(1)}_{\text{tele}}(k-1) \quad (25)$$

$$n^{(1)}_{\text{tele}}(1) = 1. \quad (26)$$

Solving this recurrence relation, we obtain

$$n^{(1)}_{\text{tele}}(k) = 8 - \frac{7}{2^{k-1}}, \quad (27)$$

which asymptotes to 8 for large k. The total number of gates used by teleportation is $2^{k+2} - 1$ times this value:

$$n_{\text{tele}}(k) = \left(2^{k+2} - 1\right)\left(8 - 14 \cdot 2^{-k}\right). \quad (28)$$

Finally, the last qubit in the distillation circuit (the one on the other half of the Bell pair) must idle while all these teleportation steps are happening. The first teleportation can be scheduled so that the $Z_k|+\rangle$ preparations are coincident with the last $\Lambda(X^q)$ operation, causing just two I gates to occur on this last qubit before the Z_{k-1} gates are optionally called. The last qubit must idle further if even a single one of the teleportations requires a Z_{k-1} corrective step. This additional idle occurs with probability $(1 - (1/2)^{2^{k+2}-1})$. We will consider a scenario in which the $Z_{k-1}|+\rangle$ states are not prepared until we know they are needed. The next teleportation iteration will incur three more I steps on the last qubit plus the time it would take to perform a Z_{k-2} correction, if needed. Since the expected fraction of original qubits requiring Z_{k-1} gates is 1/2, the probability that at least one Z_{k-2} correction is needed is $(1 - (1/2)^{(2^{k+2}-1)/2})$. Using a as a shorthand to denote $2^{k+2} - 1$, we see that the expression for the total number of identity gates incurred by

the final qubit is

$$n_{I,\,\mathrm{tele}}(k) = 2 + \left(1 - 2^{-a/2^0}\right)\left[3 + \left(1 - 2^{-a/2^1}\right) \times \left[3 + \ldots + \left(1 - 2^{-a/2^{k-2}}\right)\right]\right] \quad (29)$$

$$= 3(k-1) - \mathcal{O}(2^{-a/2^{k-2}}). \quad (30)$$

Because the error term is very small, we will neglect it and use $3(k-1)$ to represent the number of identity gates the last qubit experiences.

Putting it all together, the number of gates in the circuit depicted in Fig. 4 is

$$n^{(1)}_{\mathrm{distill}} = n_{\mathrm{pre}\ Z_k} + n_{\mathrm{tele}}(k) + n_{I,\,\mathrm{tele}}(k) + n_{\mathrm{post}\ Z_k} \quad (31)$$

$$= 2^{k+1}(k+21) - k^2 + 2k - 64 + 14 \cdot 2^{-k}. \quad (32)$$

The circuit in Fig. 4 may need be repeated if the M_X measurements fail to distill a state. Moreover, the whole process may need to be iterated to distill ultra-high-fidelity states. A subtlety is that because we did not use distilled $Z_j|+\rangle$ states for $j < k$ in the possible corrective steps required to teleport the Z_k gate, the error in each Z_k gate is not ϵ but rather some multiple of ϵ equal to the number of corrective steps required. We expect that while all the Z_k gates will have error at least ϵ, half of the remainder will have error 2ϵ because they require an additional teleportation, then a quarter of the remainder of those will have error 3ϵ, an eighth of that remainder will have error 4ϵ, and so forth until we reach the maximum number of errors which is $(k-1)$, since the S gate is error-free. The average error per Z_k gate is therefore

$$\bar{\epsilon} = \epsilon \sum_{j=1}^{k-1} j\, 2^{-(j-1)} \quad (33)$$

$$= 4\epsilon\left[1 - 2^{-k}(k+1)\right] \quad (34)$$

$$\leq 4\epsilon. \quad (35)$$

Since which qubits will suffer which multiple of ϵ in their error probability is chosen uniformly at random, this process is equivalent to the process in which each qubit independently undergoes an error process in which its error rate is chosen to be ϵ plus an additional ϵ with probability 1/2, an additional 2ϵ with probability 1/4, an additional 3ϵ with probability 1/8 and so forth, which has an expected error rate of $\bar{\epsilon}$. Because of the linearity of the trace distance, this means that the effective noise process is the same as the original one, but with ϵ replaced by $\bar{\epsilon}$. We can therefore make this replacement in the expressions for the number of repetitions t and number of iterations ℓ required. The resulting total number of gates needed to distill a $Z_k^\dagger|+\rangle$ state in our CISC architecture is

$$n_{\mathrm{distill}}(Z_k, \epsilon', \epsilon) = n^{(1)}_{\mathrm{distill}} \mathbb{E}[t(\bar{\epsilon})]\ell(C_1\epsilon', \bar{\epsilon}). \quad (36)$$

In order to compare the performance of our CISC architecture to the RISC architecture, we chose target values for ϵ' achieved at various Dawson-Nielsen iterations levels n_{DN} listed in Table I. We also chose $C_1 = C_2 = 1/2$ to balance the quality demands of distillation and the final teleportation. Our RISC and CISC results are presented side-by-side in Table I for ready comparison.

V. CONCLUSIONS

Table I summarizes our main results. A graphical way of depicting the same data is shown in Fig. 5.

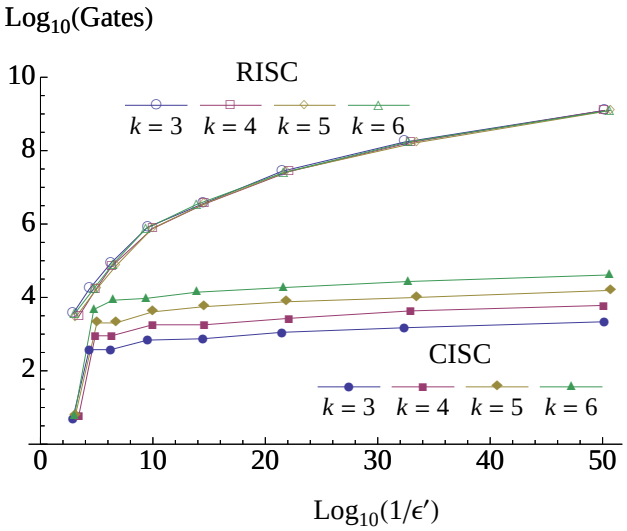

FIG. 5: Log of the number of gates required to synthesize the quantum $Z(\pi/2^k)$ gate as a function of the log of the inverse of the desired precision ϵ' for the RISC architecture described in the text and our CISC architecture.

From the log-log plot in Fig. 5, two features indicating the advantages of our quantum CISC architecture over the quantum RISC architecture are immediately apparent. First, our quantum CISC architecture uses significantly fewer gates to synthesize Z_k gates than the quantum RISC architecture does for k up to 6, a gap that grows as the precision demand grows. Second, up to a precision demand at least 10^{-50}, the number of gates required by our quantum CISC architecture scales much more modestly with the precision demand than the quantum RISC architecture does. In combination, these two features demonstrate that our quantum CISC architecture is much less resource-intensive than the quantum RISC architecture is in this regime.

The dramatic difference between the architectures at low precision demand reflects the fact that when the hardware error rate is already below this demand (*i.e.*, when $\epsilon < \epsilon'$), the only gates required by our quantum CISC architecture are those used to teleport the gate Z_k from the state $Z_k|+\rangle$ to the target state $|\psi\rangle$. The RISC architecture doesn't include the Z_k gate for $k > 2$, so it

must instead use a quantum compiling strategy to synthesize Z_k from $T|+\rangle$ states.

Our CISC architecture does have some limitations. To begin, as can be seen in Fig. 5, as k increases, even at fixed precision demand ϵ', the number of gates our CISC architecture uses increases. Eventually, at any fixed ϵ', there will be some k for which the RISC architecture uses fewer gates. However, a feature not apparent in this plot but apparent from Table II, even before this happens, the distillation threshold for our CISC architecture drops to a point below the accuracy threshold for fault-tolerant quantum computation. Using our CISC architecture beyond $k = 6$ would be foolhardy, as suddenly the distillation of encoded instructions and not the capacity of the underlying code would set the experimental hardware demands at the physical level. For this reason, we advocate using our CISC architecture up to $k = 6$, and then relying on an external quantum compiling algorithm (but with a much larger base instruction set than a quantum RISC architecture would have!) to synthesize Z_k rotations for larger k values.

We focused on synthesizing Z_k rotations for two reasons. First, numerous quantum algorithms rely on the quantum Fourier transform, which in turn is naturally decomposed into Clifford operations and Z_k rotations. We thought it was important to focus on synthesizing transformations that arise in actual algorithms rather than operations that occur only in the abstract. Second, and more significantly, we were able to find a code family, the shortened quantum Reed-Muller codes, we could leverage to create distillation protocols for Z_k rotations. The key enabling property these codes possess is *code divisibility*. With this insight, we generalized the "tri-orthogonality" condition of Bravyi and Haah [35] to a condition we call Ward's Divisibility Test, which recognizes its analogous role in classical coding theory [73]. We haven't sought codes beyond the shortened quantum Reed-Muller codes that pass Ward's Divisibility Test for admitting a Z_k-distillation protocol. However, we present and prove the correctness of this test in Appendix B in the hopes that others will find it helpful in the quest to improve quantum CISC architectures.

One of the overall messages of our work is that it is not optimal to first optimize the number of gates used to synthesize a universal instruction set and then optimize the number of universal instructions needed to synthesize a gate of interest, in this case, a Z_k gate. Instead, one can reap significant advantages by approaching this as a single optimization problem. The best conjectured asymptotic scaling when approached as two separate problems requires a number of gates that scales as $\mathcal{O}(\log^2(1/\epsilon'))$. By approaching this as a single optimization problem, one may be able to achieve $\mathcal{O}(\log(1/\epsilon'))$ for the combined process.

The resource tradeoff space for implementing quantum operations with finite discrete instruction sets is an area ripe for investigation. Beyond just minimizing the number of instructions required to approximate transformations of interest (our focus here), one might be interested in minimizing other metrics, such as the number of qubits used, the depth of the approximating quantum circuit, or the size of the approximating quantum circuit (which is its depth times the number of qubits). Depending on the task at hand, one instruction set may be more suitable than another. Investigations along these lines help us better understand the limits and capabilities of finite-instruction-set quantum information processing.

Acknowledgments

AJL and CC were supported in part by the Laboratory Directed Research and Development program at Sandia National Laboratories. Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Appendix A: Quantum Reed-Muller codes

One of the challenges in discussing quantum Reed-Muller codes is that there is not a unique definition of what a quantum Reed-Muller code is in the literature [34, 75–79]. Fortunately, there is at least a well-established definition for what a classical Reed-Muller code is. We state the definition for classical Reed-Muller codes below, confining our attention to binary codes. We refer the reader to standard texts for the definitions of supporting concepts such as Boolean monomials and $GF(2)$ [80].

Definition 1. The *rth-order binary Reed-Muller code of length 2^m*, denoted $RM(r, m)$, is the linear code over $GF(2)$ whose generator matrix is composed of row vectors corresponding to the Boolean monomials over $GF(2)^{2^m}$ of degree at most r.

As an example, the generator matrix for the $RM(1, 4)$ code is

$$G = \begin{bmatrix} 1&1&1&1&1&1&1&1&1&1&1&1&1&1&1&1 \\ 1&1&1&1&1&1&1&1&0&0&0&0&0&0&0&0 \\ 1&1&1&1&0&0&0&0&1&1&1&1&0&0&0&0 \\ 1&1&0&0&1&1&0&0&1&1&0&0&1&1&0&0 \\ 1&0&1&0&1&0&1&0&1&0&1&0&1&0&1&0 \end{bmatrix}. \quad \text{(A1)}$$

From this definition, the codespace of binary Reed-Muller codes is just the space of Boolean polynomials over $GF(2)^{2^m}$ of degree at most r. It is a minor combinatoric exercise to work out that the code $RM(r, m)$ has rank $k = \sum_{i=0}^{r} \binom{m}{i}$ and code distance $d = 2^{m-r}$. In standard coding theory notation, we say that the code

$RM(r,m)$ is an

$$[n,k,d] = \left[2^m,\ \sum_{i=0}^{r}\binom{m}{i},\ 2^{m-r}\right] \tag{A2}$$

code.

It is straightforward to work out that the dual code to $RM(r,m)$ is $RM(m-r-1,m)$. We use this to define a quantum Reed-Muller code as a CSS code composed of $RM(r,m)$ and its dual:

Definition 2. The *rth-order quantum binary Reed-Muller code of length* 2^m, denoted $QRM(r,m)$, is the CSS code [81, 82] whose defining X and Z parity check matrices are the generator matrices for $RM(r,m)$ and its dual $RM(m-r-1,m)$ respectively.

Notice that in this definition, somewhat confusingly, the quantum parity-check matrices are formed from classical *generator* matrices, not classical parity-check matrices.

We are most interested in the *shortened* quantum binary Reed-Muller codes, which we denote by $\overline{QRM}(r,m)$. These codes are formed by shortening each of the binary Reed-Muller codes from which it is formed. The process of shortening first punctures a code by removing a bit on which only row of the generator matrix has support and then expurgates it by removing the row in the generator matrix that had support on that bit. For the Reed-Muller codes, this corresponds to removing the first row and last column of the generator matrix when presented in standard form, as in Eq. (A1). In essence, shortening a Reed-Muller code restricts the space of Boolean polynomials defining the code to those which have no constant term and which also satisfy $p(0)=0$. An equivalent way of characterizing the shortened Reed-Muller code is as the even subcode of the punctured Reed-Muller code. The parameters of the resulting quantum code are $[\![2^m-1,1]\!]$. Code parameters for small Reed-Muller codes, their duals, and their shortened quantum construct are listed in Table III. Notice that the length of the code n does not uniquely specify which shortened quantum Reed-Muller code one is referring to for $n>15$.

(r,m)	$(m-r-1,m)$	$[n,k,d]$ primal	$[n,k,d]$ dual	$[\![n,k]\!]$
(0,1)	(0,1)	[2,1,2]	[2,1,2]	∅
(0,2)	(1,2)	[4,1,4]	[4,3,2]	∅
(0,3)	(2,3)	[8,1,8]	[8,7,2]	∅
(1,3)	(1,3)	[8,4,4]	[8,4,4]	$[\![7,1]\!]$
(0,4)	(3,4)	[16,1,16]	[16,15,2]	∅
(1,4)	(2,4)	[16,5,8]	[16,11,4]	$[\![15,1]\!]$
(0,5)	(4,5)	[32,1,32]	[32,31,2]	∅
(1,5)	(3,5)	[32,6,16]	[32,26,4]	$[\![31,1]\!]$
(2,5)	(2,5)	[32,16,8]	[32,16,8]	$[\![31,1]\!]$
(0,6)	(5,6)	[64,1,64]	[64,63,2]	∅
(1,6)	(4,6)	[64,7,32]	[64,57,4]	$[\![63,1]\!]$
(2,6)	(3,6)	[64,22,32]	[64,42,8]	$[\![63,1]\!]$

TABLE III: Parameters for (primal) Reed-Muller $R(r,m)$ codes, their duals $R(m-r-1,1)$, and their CSS-combined shortened quantum versions $\overline{QRM}(r,m)$ for small values. Shortened $R(0,m)$ codes have no X generator, so the resulting quantum codes are just classical codes; they are referred to by ∅ in the table.

Appendix B: Criteria for a code to admit transversal $Z(\pi/2^k)$ rotations

The shortened quantum Reed-Muller codes $\overline{QRM}(1,k+2)$ admit a transversal implementation of Z_k by applying $Z_k^\dagger$ to each qubit in the code independently. This result follows, *e.g.* from arguments made by Campbell *et al.* in Ref. [79]. Another way to see this is to note that these codes obey Theorem 1 below. We offer this alternative approach because it may be generalizable in a way that others could use to find more efficient codes that admit Z_k transversally. It also relies on a lemma (Lemma 1) that naturally generalizes an otherwise unusual criterion of "tri-orthogonality" noted by Bravyi and Haah [35] for the $\overline{QRM}(1,4)$ code. We believe that this Lemma, which we call *Ward's Divisibility Test*, makes better contact with the classical coding theory literature.

Theorem 1. A quantum $[\![n,1]\!]$ CSS code [81, 82] with stabilizer generators defined by the parity check matrix $H=\mathrm{diag}(H^X,H^Z)$ via

$$S_i^X := \bigotimes_{j=1}^{n} X^{H_{ij}^X} \qquad S_i^Z := \bigotimes_{j=1}^{n} Z^{H_{ij}^Z}, \tag{B1}$$

where H^X has rows $v_1,\ldots v_{k+2}$, implements $(Z_k)^a$ transversally if

$$\mathrm{wt}\left(v_{\sigma(1)}\cdots v_{\sigma(j)}\right) \equiv 0 \bmod 2^{k+2-j} \tag{B2}$$

for all $1\le j\le k+2$ and all $\sigma\in\Sigma_j$, and

$$n \equiv a \bmod 2^{k+1}, \tag{B3}$$

where '$\otimes$' denotes the tensor product, 'wt' denotes the Hamming weight of a binary vector, 'Σ_j' denotes the permutation group on j items, and '$v_1\cdots v_j$' denotes the componentwise product of $v_1,\ldots,v_j$.

When a in this Theorem is odd, $\gcd(a,2^{k+1})=1$, which means we can use an algorithm like the extended Euclidean algorithm [83] to efficiently find numbers x and y such that $ax+2^{k+1}y=1$. Iterating $(Z_k)^a$ x times results in a conditional phase of $\pi(1-2^{k+1}y)/2^k \cong \pi/2^k$; in other words, $(Z_k)^{ax}\cong Z_k$ when a is odd.

Condition (B2) generalizes the tri-orthogonality condition of Bravyi and Haah [35] into a kind of $(k+1)$-orthogonality condition. More fundamentally, we want

the classical linear code generated by H^X to be a code in which every codeword has a Hamming weight divisible by 2^{k+1}. Ward studied such *divisible codes* in depth and one of his results is that 2^{k+1}-divisibility is testable by the condition of Eq. (B2) [73]. More explicitly, Ward's Divisibility Test is captured by Lemma 1 below. (Ward's result is actually more general; we use a version specialized to the binary case, as noted by Proposition 4.2 in Ref. [74].)

Lemma 1 (Ward's Divisibility Test [73])**.** The binary linear code with generator matrix H^X whose row vectors are $v_1, \ldots, v_{k+2}$ is divisible by 2^{k+1} if and only if

$$2^{k+2-j} \Big| \mathrm{wt}(v_{\sigma(1)} \cdots v_{\sigma(j)}) \tag{B4}$$

for all $1 \leq j \leq k+1$ and all permutations $\sigma \in \Sigma_j$.

While Ward's Divisibility Test has the advantage of being an explicit algorithm for testing divisibility, it is not particularly efficient, as it takes a time that is exponential in k to execute. For codes with a high degree of structure, such as the shortened $\overline{RM}(1, k+2)$ Reed-Muller codes, demonstrating 2^{k+1} divisibility is much simpler, as noted in Ref. [74].

Proof of Theorem 1. By Ward's Divisibility Test, every vector v in the rowspan $\mathcal{L}$ of H^X has a Hamming weight divisible by 2^{k+1}. Since the logical $|0\rangle$ for the code is $|\overline{0}\rangle := \sum_{v \in \mathcal{L}} |v\rangle$ (ignoring normalization), the action of transversal Z_k on $|\overline{0}\rangle$ is

$$\begin{aligned} Z_k^{\otimes n} |\overline{0}\rangle &= \sum_{v \in \mathcal{L}} Z_k^{\otimes n} |v\rangle && \text{(B5)} \\ &= \sum_{v \in \mathcal{L}} \left(e^{i\pi/2^k}\right)^{|v|} |v\rangle && \text{(B6)} \\ &= \sum_{v \in \mathcal{L}} |v\rangle && \text{(B7)} \\ &= |\overline{0}\rangle. && \text{(B8)} \end{aligned}$$

Similarly, using Eq. (B3), the action of transversal Z_k on (unnormalized) $|\overline{1}\rangle = \overline{X}|\overline{0}\rangle$ is

$$\begin{aligned} Z_k^{\otimes n} |\overline{1}\rangle &= Z_k^{\otimes n} \overline{X} |\overline{0}\rangle && \text{(B9)} \\ &= \sum_{v \in \mathcal{L}} Z_k^{\otimes n} \overline{X} |v\rangle && \text{(B10)} \\ &= \sum_{v \in \mathcal{L}} Z_k^{\otimes n} |v \oplus \mathbf{1}\rangle && \text{(B11)} \\ &= \sum_{v \in \mathcal{L}} \left(e^{i\pi/2^k}\right)^{n-|v|} |v \oplus \mathbf{1}\rangle && \text{(B12)} \\ &= \sum_{v \in \mathcal{L}} \left(e^{i\pi a/2^k}\right) |v \oplus \mathbf{1}\rangle && \text{(B13)} \\ &= e^{i\pi a/2^k} |\overline{1}\rangle, && \text{(B14)} \end{aligned}$$

where $\mathbf{1} := (1, \ldots, 1)$ denotes the all-ones vector, whose appearance comes from the fact that up to local qubit basis changes, $\overline{X} = X^{\otimes n}$ for all CSS codes. These actions of $Z^{\otimes n}$ replicate $(Z_k)^a$ on the logical basis, and therefore Z_k implements $(Z_k)^a$ transversally. □

[1] E. Farhi and S. Gutmann, *Analog analogue of a digital quantum computation*, Phys. Rev. A **57**, 2403 (1998), doi:10.1103/PhysRevA.57.2403, arXiv:quant-ph/9612026.

[2] D. Aharonov, W. van Dam, J. Kempe, Z. Landau, S. Lloyd, and O. Regev, *Adiabatic quantum computation is equivalent to standard quantum computation*, in *Proceedings of the 45th IEEE Symposium on Foundations of Computer Science*, edited by D. Azada, IEEE (IEEE Press, Los Alamitos, CA, Oct. 17–19, Rome, IT, 2004), pp. 42–51, ISBN 0-7695-2228-9, doi:10.1109/FOCS.2004.8, arXiv:quant-ph/0405098.

[3] A. Mizel, D. A. Lidar, and M. Mitchell, *Simple proof of equivalence between adiabatic quantum computation and the circuit model*, Phys. Rev. Lett. **99**, 070502 (2007), doi:10.1103/PhysRevLett.99.070502, arXiv:quant-ph/0609067.

[4] A. M. Childs, E. Farhi, and J. Preskill, *Robustness of adiabatic quantum computation*, Phys. Rev. A **65**, 012322 (2001), doi:10.1103/PhysRevA.65.012322, arXiv:quant-ph/0108048.

[5] J. Åberg, D. Kult, and E. Sjöqvist, *Robustness of the adiabatic quantum search*, Phys. Rev. A **71**, 060312(R) (2005), doi:10.1103/PhysRevA.71.060312, arXiv:quant-ph/0412124.

[6] J. Åberg, D. Kult, and E. Sjöqvist, *The quantum adiabatic search with decoherence in the instantaneous energy eigenbasis*, Phys. Rev. A **72**, 042317 (2005), doi:10.1103/PhysRevA.72.042317, arXiv:quant-ph/0507010.

[7] M. S. Sarandy and D. A. Lidar, *Adiabatic quantum computation in open systems*, Phys. Rev. Lett. **95**,

250503 (2005), doi:10.1103/PhysRevLett.95.250503, arXiv:quant-ph/0502014.

[8] J. Roland and N. J. Cerf, *Noise resistance of adiabatic quantum computation using random matrix theory*, Phys. Rev. A **71**, 032330 (2005), doi:10.1103/PhysRevA.71.032330, arXiv:quant-ph/0409127.

[9] S. Ashhab, J. R. Johansson, and F. Nori, *Title: Decoherence in a scalable adiabatic quantum computer*, Phys. Rev. A **74**, 052330 (2006), doi:10.1103/PhysRevA.74.052330, arXiv:quant-ph/0608212.

[10] F. Gaitan, *Simulation of quantum adiabatic search in the presence of noise*, Int. J. Quant. Inf. **4**, 843 (2006), doi:10.1142/S0219749906002213, arXiv:quant-ph/0601116.

[11] M. Tiersch and R. Schützhold, *Non-Markovian decoherence in the adiabatic quantum search algorithm*, Phys. Rev. A **75**, 062313 (2007), doi:10.1103/PhysRevA.75.062313, arXiv:quant-ph/0608123.

[12] M. H. S. Amin, D. V. Averin, and J. A. Nesteroff, *Decoherence in adiabatic quantum computation*, Phys. Rev. A **79**, 022107 (2009), doi:10.1103/PhysRevA.79.022107, arXiv:0708.0384.

[13] M. H. S. Amin, C. J. S. Truncik, and D. V. Averin, *Role of single qubit decoherence time in adiabatic quantum computation*, Phys. Rev. A **80**, 022303 (2009), doi:10.1103/PhysRevA.80.022303, arXiv:0803.1196.

[14] I. de Vega, M. C. Bañuls, and A. Pérez, *Effects of dissipation on an adiabatic quantum search algorithm*, New J. Phys. **12**, 123010 (2010), doi:10.1088/1367-2630/12/12/123010, arXiv:1006.0461.

[15] P. W. Shor, *Scheme for reducing decoherence in quantum computer memory*, Phys. Rev. A **52**, R2493 (1995), doi:10.1103/PhysRevA.52.R2493.

[16] P. W. Shor, *Fault-tolerant quantum computation*, in *Proceedings of the 37th Annual Symposium on Foundations of Computer Science*, edited by R. S. Sipple, IEEE (IEEE Press, Los Alamitos, CA, 14–16 Oct. 1996, Burlington, VT, USA, 1996), pp. 56–65, ISBN 0-8186-7594-2, doi:10.1137/S0097539795293172, arXiv:quant-ph/9605011.

[17] D. Aharonov and M. Ben-Or, *Fault tolerant quantum computation with constant error*, in *Proceedings of the Twenty-Ninth Annual ACM Symposium on the Theory of Computing*, edited by F. T. Leighton and P. Shor (ACM Press, New York, El Paso, TX, USA, 1997), pp. 176–188, ISBN 0-89791-888-6, See also extended version [18]., doi:10.1145/258533.258579, arXiv:quant-ph/9611025.

[18] D. Aharonov and M. Ben-Or, *Fault tolerant quantum computation with constant error rate* (1999), See also condensed version [17]., arXiv:quant-ph/9906129.

[19] A. Y. Kitaev, *Quantum computations: algorithms and error correction*, Russian Math. Surveys **52**, 1191 (1997), doi:10.1070/RM1997v052n06ABEH002155.

[20] A. M. Steane, *Active stabilization, quantum computation, and quantum state synthesis*, Phys. Rev. Lett. **78**, 2252 (1997), doi:10.1103/PhysRevLett.78.2252, arXiv:quant-ph/9611027.

[21] E. Knill, R. Laflamme, and W. H. Zurek, *Resilient quantum computation: Error models and thresholds*, Proc. Roy. Soc. London A **454**, 365 (1998), arXiv:quant-ph/9702058, URL http://www.jstor.org/stable/53171.

[22] J. Preskill, *Fault-tolerant quantum computation*, in *Introduction to Quantum Computation and Information*, edited by H.-K. Lo, T. Spiller, and S. Popescu (World Scientific, Singapore / River Edge, NJ, 1998), chap. 8, pp. 213–269, ISBN 9-810-24410-X, arXiv:quant-ph/9712048.

[23] J. Preskill, *Reliable quantum computers*, Proc. Roy. Soc. London A **454**, 385 (1998), arXiv:quant-ph/9705031, URL http://www.jstor.org/stable/53172.

[24] A. Y. Kitaev, *Quantum error correction with imperfect gates*, in *Proceedings of the Third International Conference on Quantum Communication, Computing and Measurement*, edited by O. Hirota, A. S. Holevo, and C. M. Caves (Plenum Press, New York, 1997).

[25] E. Dennis, A. Kitaev, A. Landahl, and J. Preskill, *Topological quantum memory*, J. Math. Phys. **43**, 4452 (2002), doi:10.1063/1.1499754, arXiv:quant-ph/0110143.

[26] R. Raussendorf, J. Harrington, and K. Goyal, *Topological fault-tolerance in cluster state quantum computation*, New J. Phys. **9**, 199 (2007), doi:10.1088/1367-2630/9/6/199, arXiv:quant-ph/0703143.

[27] A. G. Fowler, A. M. Stephens, and P. Groszkowski, *High-threshold universal quantum computation on the surface code*, Phys. Rev. A **80**, 052312 (2009), doi:10.1103/PhysRevA.80.052312, arXiv:0803.0272.

[28] S. P. Jordan, E. Farhi, and P. W. Shor, *Error correcting codes for adiabatic quantum computation*, Phys. Rev. A **74**, 052322 (2006), doi:10.1103/PhysRevA.74.052322, arXiv:quant-ph/0512170.

[29] D. A. Lidar, *Towards fault tolerant adiabatic quantum computation* (2007), arXiv:0707.0021.

[30] G. A. Paz-Silva, A. T. Rezakhani, J. M. Dominy, and D. A. Lidar, *Zeno effect for quantum computation and control*, Phys. Rev. Lett. **108**, 080501 (2012), doi:10.1103/PhysRevLett.108.080501, arXiv:1104.5507.

[31] K. C. Young and M. Sarovar, *Equivalence and limitations of error suppression techniques for adiabatic quantum computing* (2012), arXiv:1208.6371.

[32] C. S. Ahn, *Extending quantum error correction: new continuous measurement protocols and improved fault-tolerant overhead*, Ph.D. thesis, Caltech (2004).

[33] B. Eastin and E. Knill, *Restrictions on transversal encoded quantum gate sets*, Phys. Rev. Lett. **102**, 110502 (2009), doi:10.1103/PhysRevLett.102.110502, arXiv:0811.4262.

[34] S. Bravyi and A. Kitaev, *Universal quantum computation with ideal Clifford gates and noisy ancillas*, Phys. Rev. A **71**, 022316 (2005), doi:10.1103/PhysRevA.71.022316, arXiv:quant-ph/0403025.

[35] S. Bravyi and J. Haah, *Magic state distillation with low overhead* (2012), arXiv:1209.2426.

[36] N. C. Jones, J. D. Whitfield, P. L. McMahon, M.-H. Yung, R. Van Meter, A. Aspuru-Guzik, and Y. Yamamoto, *Simulating chemistry efficiently on fault-tolerant quantum computers* (2012), arXiv:1204.0567.

[37] H. Bombin and M. A. Martin-Delgado, *Exact topological quantum order in $d = 3$ and beyond: Branyons and brane-net condensates*, Phys. Rev. B **75**, 075103 (2007), doi:10.1103/PhysRevB.75.075103,

arXiv:cond-mat/0607736.

[38] A. J. Landahl, J. T. Anderson, and P. R. Rice, *Fault-tolerant quantum computing with color codes* (2011), arXiv:1108.5738.

[39] A. W. Harrow, B. Recht, and I. L. Chuang, *Efficient discrete approximations of quantum gates*, J. Math. Phys. **43**, 4445 (2002), doi:10.1063/1.1495899, arXiv:quant-ph/0111031.

[40] V. Kliuchnikov, D. Maslov, and M. Mosca, *Asymptotically optimal approximation of single qubit unitaries by Clifford and T circuits using a constant number of ancillary qubits* (2012), arXiv:1212.0822.

[41] C. M. Dawson and M. A. Nielsen, *The Solovay-Kitaev algorithm*, Quant. Inf. Comp. **6**, 81 (2006), arXiv:quant-ph/0505030, URL http://www.rintonpress.com/xqic6/qic-6-1/081-095.pdf.

[42] M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge University Press, Cambridge, 2000), ISBN 0-521-63235-8 (Hardback), 0-521-63503-9 (Paperback).

[43] D. Gottesman, *The Heisenberg representation of quantum computers*, in *Group22: Proceedings of the XXII International Colloquium on Group Theoretical Methods in Physics*, edited by S. P. Corney, R. Delbourgo, and P. D. Jarvis (International Press, Cambridge, MA, 13–17 Jul. 1998, Hobart, Australia, 1999), pp. 32–43, arXiv:quant-ph/9807006.

[44] J. T. Anderson, *On the power of reusable magic states* (2012), arXiv:1205.0289.

[45] M. F. Sacchi, *Optimal discrimination of quantum operations*, Phys. Rev. A **71**, 062340 (2005), doi:10.1103/PhysRevA.71.062340, arXiv:quant-ph/0505183.

[46] J. Watrous, *Semidefinite programs for completely bounded norms* (2009), arXiv:0901.4709.

[47] R. Solovay, *Lie groups and quantum circuits* (2000), MSRI presentation at http://www.msri.org/publica-tions/ln/msri/2000/qcomputing/solovay/1/. According to Ref. [41], the proof was announced by Solovay in 1995 on a virtual reading group e-mail discussion list.

[48] A. G. Fowler, *Towards larg-scale quantum computation*, Ph.D. thesis, University of Melbourne (2005), arXiv:quant-ph/0506126.

[49] P. T. Trung, R. Van Meter, and C. Horsman, *Optimising the Solovay-Kitaev algorithm* (2012), arXiv:1209.4139.

[50] A. Y. Kitaev, *Quantum measurements and the abelian stabilizer problem* (1995), arXiv:quant-ph/9511026.

[51] R. Cleve, A. Ekert, C. Macchiavello, and M. Mosca, *Quantum algorithms revisited*, Proc. Roy. Soc. London A **454**, 339 (1998), arXiv:quant-ph/9708016, URL http://www.jstor.org/stable/53169.

[52] A. Yu. Kitaev, A. Shen, and M. N. Vyalyi, *Classical and Quantum Computation*, vol. 47 of *Graduate Studies in Mathematics* (American Mathematical Society, Providence, RI, 2002), ISBN 0-821-82161-X, Translated from the Russian edition by Lester J. Senechal.

[53] N. Isailovic, M. Whitney, Y. Patel, and J. Kubiatowicz, *Running a quantum circuit at the speed of data*, IEEE (IEEE Press, Los Alamitos, 21–25 Jun., 2008, Beijing, China, 2008), p. 177, doi:10.1109/ISCA.2008.5, arXiv:0804.4725.

[54] G. Duclos-Cianci and K. M. Svore, *A state distillation protocol to implement arbitrary single-qubit rotations* (2012), arXiv:1201.1980.

[55] R. B. McDonald and H. G. Katzgraber, *Genetic braid optimization: A heuristic approach to compute quasiparticle braids* (2012), arXiv:1211.7359.

[56] A. Fowler, *Constructing arbitrary Steane code single logical qubit fault-tolerant gates*, Quant. Inf. Comp. **11**, 867 (2011), arXiv:quant-ph/0411206, URL http://www.rintonpress.com/xxqic11/qic-11-910/0867-0873.pdf.

[57] M. Amy, D. Maslov, M. Mosca, and M. Roetteler, *A meet-in-the-middle algorithm for fast synthesis of depth-optimal quantum circuits* (2012), arXiv:1206.0758.

[58] V. Kliuchnikov, D. Maslov, and M. Mosca, *Fast and efficient exact synthesis of single qubit unitaries generated by Clifford and T gates* (2012), arXiv:1206.5236.

[59] A. Bocharov and K. Svore, *A depth-optimal canonical form for single-qubit quantum circuits* (2012), arXiv:1206.3223.

[60] A. Gilchrist, N. K. Langford, and M. A. Nielsen, *Distance measures to compare real and ideal quantum processes*, Phys. Rev. A **71**, 062310 (2005), doi:10.1103/PhysRevA.71.062310, arXiv:quant-ph/0408063.

[61] S. Aaronson and D. Gottesman, *Improved simulation of stabilizer circuits*, Phys. Rev. A **70**, 052328 (2004), doi:10.1103/PhysRevA.70.052328, arXiv:quant-ph/0406196.

[62] S. Anders and H. J. Briegel, *Fast simulation of stabilizer circuits using a graph-state representation*, Phys. Rev. A **73**, 022334 (2006), doi:10.1103/PhysRevA.73.022334, arXiv:quant-ph/0504117.

[63] B. W. Reichardt, *Quantum universality from magic states distillation applied to CSS codes*, Quant. Inf. Proc. **4**, 251 (2005), doi:10.1007/s11128-005-7654-8, arXiv:quant-ph/0411036.

[64] T. Jochym-O'Connor, Y. Yu, B. Helou, and R. Laflamme, *The robustness of magic state distillation against errors in Clifford gates* (2012), arXiv:1205.6715.

[65] E. Knill, *Fault-tolerant postselected quantum computation: Schemes* (2004), arXiv:quant-ph/0402171.

[66] B. W. Reichardt, *The quantum adiabatic optimization algorithm and local minima*, in *Proceedings of the Thirty-Sixth Annual ACM Symposium on the Theory of Computing*, edited by L. L. Larmore and M. X. Goemans (ACM Press, New York, 13–15 Jun. 2004, Chicago, IL, USA, 2004), pp. 502–510, ISBN 1-58113-852-0, doi:10.1145/1007352.1007428.

[67] B. W. Reichardt, *Quantum universality by state distillation* (2006), arXiv:quant-ph/0608085.

[68] A. M. Meier, B. Eastin, and E. Knill, *Magic-state distillation with the four-qubit code* (2012), arXiv:1204.4221.

[69] N. C. Jones, R. V. Meter, A. G. Fowler, P. L. McMahon, J. Kim, T. D. Ladd, and Y. Yamamoto, *Layered architecture for quantum computing*, Phys. Rev. X **2**, 031007 (2012), doi:10.1103/PhysRevX.2.031007, arXiv:1010.5022.

[70] A. G. Fowler, M. Mariantoni, J. M. Martinis, and A. N. Cleland, *Surface codes: Towards practical large-scale quantum computation*, Phys. Rev. A **86**, 032324 (2012), doi:10.1103/PhysRevA.86.032324, arXiv:1208.0928.

[71] A. G. Fowler and S. J. Devitt, *A bridge to lower overhead quantum computation* (2012), arXiv:1209.0510.

[72] D. Gottesman, *Stabilizer codes and quantum error correction*, Ph.D. thesis, Caltech (1997),

arXiv:quant-ph/9705052.

[73] H. N. Ward, *Weight polarization and divisibility*, Discrete Math. **83**, 315 (1990), doi:10.1016/0012-365X(90)90015-A.

[74] X. Liu, *On divisible codes over finite fields*, Ph.D. thesis, Caltech (2006), URL http://resolver.caltech.edu/CaltechETD:etd-05252006-010331.

[75] A. Steane, *Quantum Reed-Muller codes* (1996), arXiv:quant-ph/9608026.

[76] L. Zhang and I. Fuss, *Quantum Reed-Muller codes* (1997), arXiv:quant-ph/9703045.

[77] J. Preskill, *Lecture notes for Caltech Ph 219: Quantum Information and Computation* (1998), URL http://www.theory.caltech.edu/~preskill/ph219/.

[78] P. Sarvepalli and A. Klappenecker, *Nonbinary quantum Reed-Muller codes*, IEEE (IEEE Press, 4–9 Sep. 2005, Adelaide, Australia, 2005), pp. 1023–1027, doi:10.1109/ISIT.2005.1523494, arXiv:quant-ph/0502001.

[79] E. T. Campbell, H. Anwar, and D. E. Browne, *Magic state distillation in all prime dimensions using quantum Reed-Muller codes* (2012), arXiv:1205.3104.

[80] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*, vol. 16 of *North-Holland mathematical library* (North-Holland, New York, 1977), ISBN 0-444-85009-0.

[81] A. R. Calderbank and P. W. Shor, *Good quantum error-correcting codes exist*, Phys. Rev. A **54**, 1098 (1996), doi:10.1103/PhysRevA.54.1098, arXiv:quant-ph/9512032.

[82] A. M. Steane, *Simple quantum error-correcting codes*, Phys. Rev. A **54**, 4741 (1996), doi:10.1103/PhysRevA.54.4741, arXiv:quant-ph/9605021.

[83] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms* (MIT Press and McGraw-Hill, 2001), ISBN 0-262-03293-7.

[84] How errors propagate in the injection process is an understudied problem in our opinion. However, we will not consider this issue here because we are abstracting away the details of quantum error correcting codes in our analysis.